

Academia de Studii Economice din București

Facultatea de Cibernetică, Statistică și Informatică Economică

Specializarea Informatică Economică

Proiect Calitate și Testare Software

”Gestiunea unor tipuri de vehicule”

Realizat de:

ISAR Cosmina, Grupa 1073, Seria B

București, 2016

Cuprins

[Definirea și justificarea pattern-urilor implementate 1](#_Toc452676793)

[1. Builder 1](#_Toc452676794)

[2. Simple Factory 2](#_Toc452676795)

[3. Adapter (de obiecte) 3](#_Toc452676796)

[4. Facade 4](#_Toc452676797)

[5. Decorator 5](#_Toc452676798)

[6. Proxy 6](#_Toc452676799)

[7. State 7](#_Toc452676800)

[8. Memento 8](#_Toc452676801)

[Definirea și detalierea metodelor testate prin Unit Testing 9](#_Toc452676802)

[1. Clasa Vehicul 9](#_Toc452676803)

[1.1. Interfață comună pentru obiectele de tip Vehicul 9](#_Toc452676804)

[1.2. Adăugare componentă preț obiectelor de tip Vehicul 9](#_Toc452676805)

[1.3. Adăugare componentă capacitate cilindrică obiectelor de tip Vehicul 9](#_Toc452676806)

[2. Clasa GestiuneInformatiiLansariAuto 10](#_Toc452676807)

[2.1. Adăugare informație privind lansările auto 10](#_Toc452676808)

[2.2. Restaurare informație anterioare 10](#_Toc452676809)

[Definirea și descrierea Test Case-urilor 10](#_Toc452676810)

[1. TestCaseConstructorVehicul 10](#_Toc452676811)

[1.1. Verificarea constructorului pentru parametri cu valori normale 10](#_Toc452676812)

[1.2. Verificarea constructorului pentru parametrul marca = null 10](#_Toc452676813)

[1.3. Verificarea constructorului pentru parametrul model = null 11](#_Toc452676814)

[1.4. Verificarea constructorului pentru ambi parametri = null 11](#_Toc452676815)

[2. TestCaseAdaugarePret 11](#_Toc452676816)

[2.1. Verificarea componentei de adaugare pret pentru parametru cu valori normale 11](#_Toc452676817)

[2.2. Verificarea componentei de adaugare pret pentru parametru cu valoare egală cu 0 12](#_Toc452676818)

[2.3. Verificarea componentei de adaugare pret pentru parametru cu valoare mai mică decât valoarea minimă acceptată 12](#_Toc452676819)

[2.4. Verificarea componentei de adaugare pret pentru parametru cu valoare mai mare decât valoarea maximă acceptată 12](#_Toc452676820)

[3. TestCaseAdaugareCapacitateCilindrica 13](#_Toc452676821)

[3.1. Verificarea componentei de adaugare capacitate cilindrică pentru parametru cu valori normale 13](#_Toc452676822)

[3.2. Verificarea componentei de adaugare capacitate cilindrică pentru parametru cu valoare egală cu 0 13](#_Toc452676823)

[3.3. Verificarea componentei de adaugare capacitate cilindrică pentru parametru cu valoare mai mică decât valoarea minimă acceptată 13](#_Toc452676824)

[3.4. Verificarea componentei de adaugare capacitate cilindrică pentru parametru cu valoare mai mare decât valoarea maximă acceptată 14](#_Toc452676825)

[4. TestCaseListaInformatiiSalvate 14](#_Toc452676826)

[4.1. Verificare metodă de adăugare a unei noi informații privind lista cu informațiile aferente lansărilor auto 14](#_Toc452676827)

[4.2. Verificarea metodei de restaurare a unei informații anterioare din lista cu informațiile aferente lansărilor auto 14](#_Toc452676828)

[Definirea și descrierea Test Suite-urilor 15](#_Toc452676829)

[1. TestSuiteImbunatatiriVehicul 15](#_Toc452676830)

[2. TestSuiteVehicul 15](#_Toc452676831)

[Descrierea sumară a funcțiilor aplicației cu referire la pattern-uri și la metodele testate 16](#_Toc452676832)

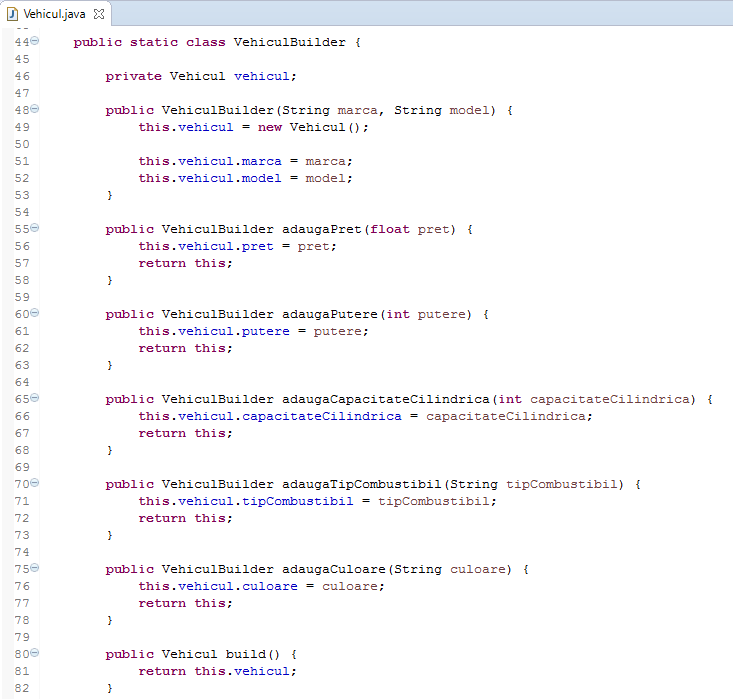
[Bibliografie 16](#_Toc452676833)

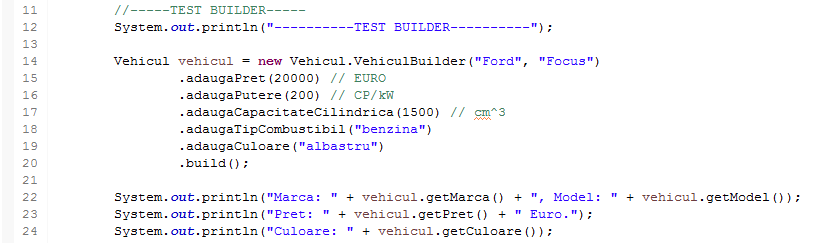
# Definirea și justificarea pattern-urilor implementate

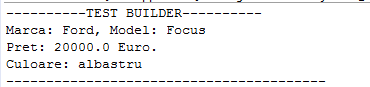
## Builder

Clase utilizate: Vehicul.

Utilizare: S-a recurs la implementarea Builder-ului, deoarece clasa Vehicul trebuie să construiacă obiecte complexe printr-un mecanism care este independent de procesul de realizare a obiectelor. Astfel interfata comună obiectelor va fi formată din cele mai relevante două atribute (marcă, model), urmând ca celelalte componente să fie adăugate în funcție de necesitate.



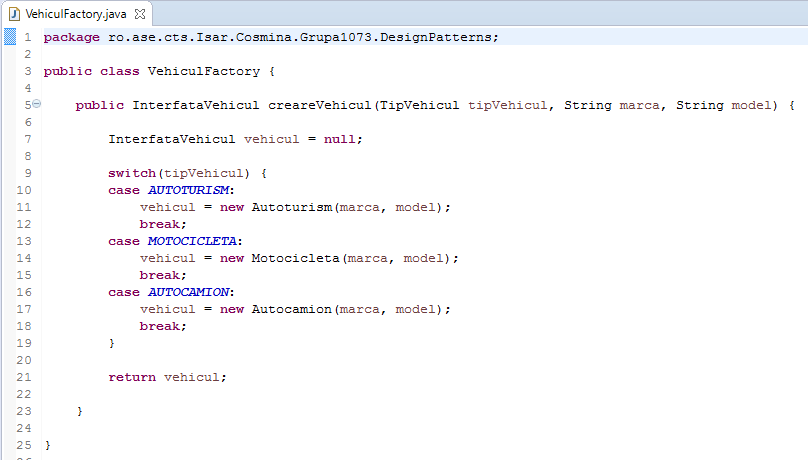


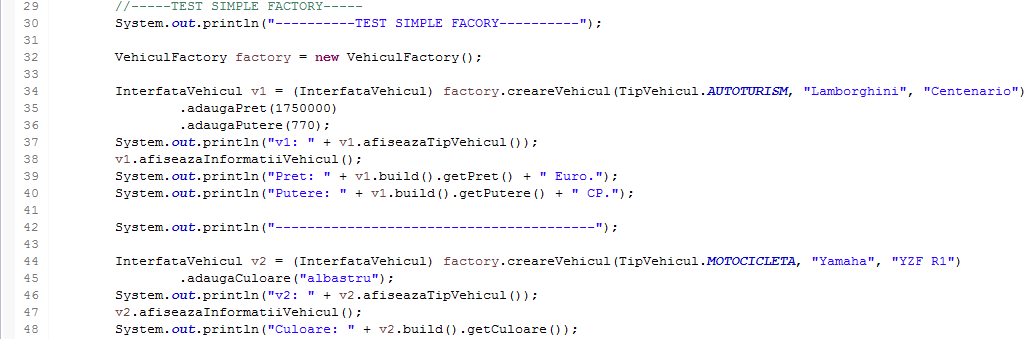


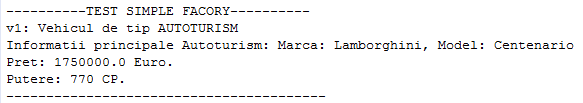
## Simple Factory

Clase utilizate: Vehicul, InterfataVehicul, TipVehicul, Autoturism, Motocicleta, Autocamion, VehiculFactory.

Utilizare: Deoarece s-a dorit extinderea clasei Vehicul prin adăugarea de noi tipuri concrete de obiecte (Autoturism, Motocicletă, Autocamion) fără a afecta codul exitent, s-a recurs la implementarea acestui Desing Pattern.



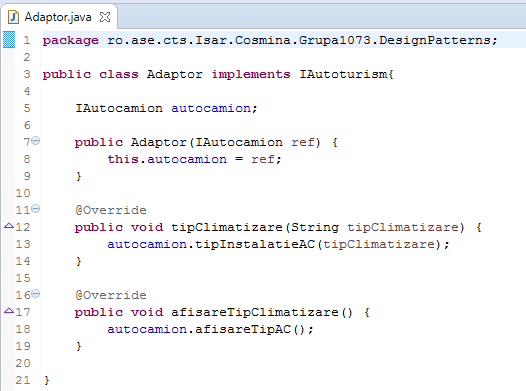


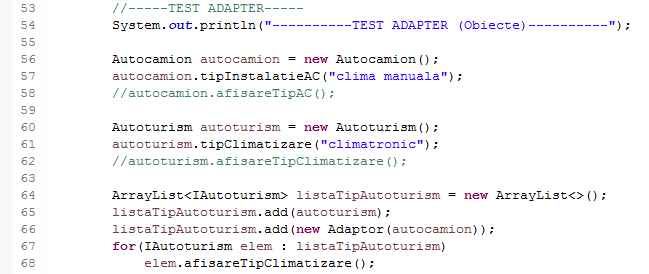


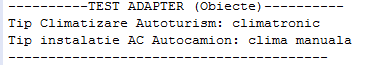
## Adapter (de obiecte)

Clase utilizate: Autoturism, IAutoturism, Autocamion, IAutocamion, Adaptor.

Utilizare: Scopul implementării Adaper-ului a fost acela de a crea o interfață comună (Adaptor) între IAutoturism și IAutocamion, adaptând interfața clasei exitente la cea a clasei din noul context. Astfel, s-a dorit ca IAutocamion să fie adaptat la o nouă interfață și anume IAutoturism.



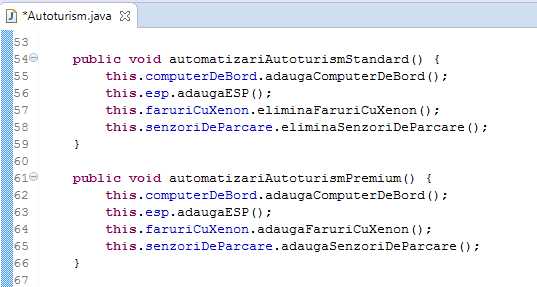


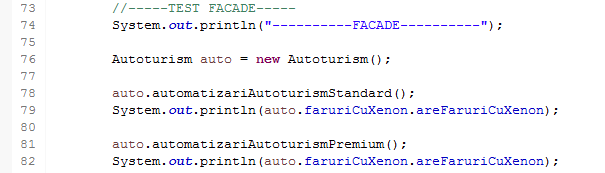


## Facade

Clase utilizate: Autoturism, ComputerDeBord, ESP, FaruriCuXenon, SenzoriDeParcare.

Utilizare: Deoarece s-a dorit dezvoltarea unei soluții software pentru managementul unui Autoturism, utilizând un număr mare de clase și gestionarea acestora pentru crearea a unor tipuri de automatizări (automatizăriAutoturismStandard() și automatizăriAutoturismPremium()), s-a recurs la implementarea acestui Design Pattern.



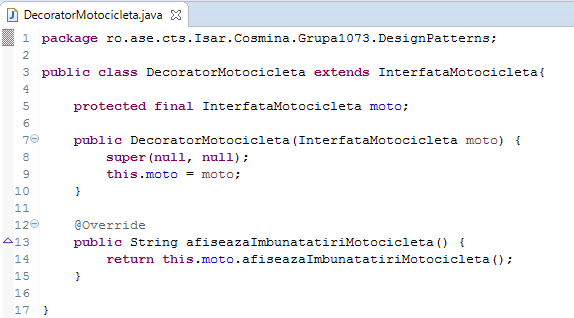


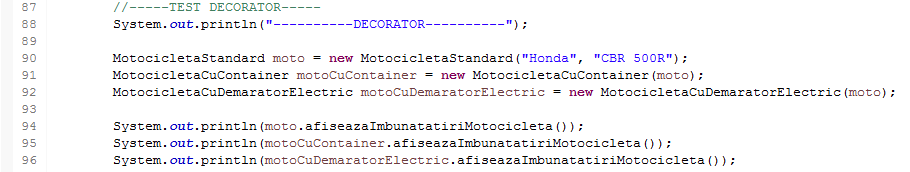
https://i.gyazo.com/c32d39c975f914d4cd061e7c370df357.png

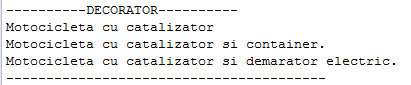
## Decorator

Clase utilizate: Motocicleta, InterfataMotocicleta, MotocicletaStandard, DecoratorMotocicleta, MotocicletaCuContainer, MotocicletaCuDemaratorElectric.

Utilizare: S-a implementat Decorator-ul, deoarece s-a dorit extinderea la run-time a funcționalității unor obiecte de tip Motociletă. Astfel, InterfataMotocicleta definește interfața obiectelor ce pot fi decorate, MotocicletaStandard surprinde obiectele ce pot fi decorate, DecoratorMotocicleta definește o interfață comună claselor decorate (MotocicletaCuContainer, MotocicletaCuDemaratorElectric).



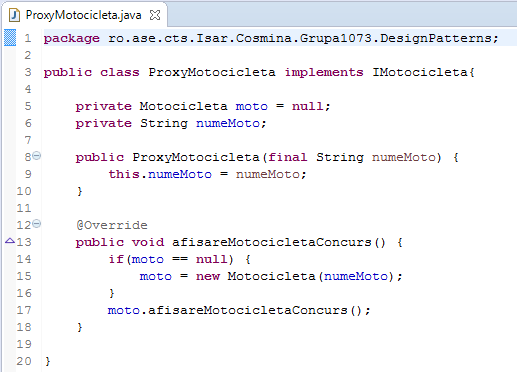


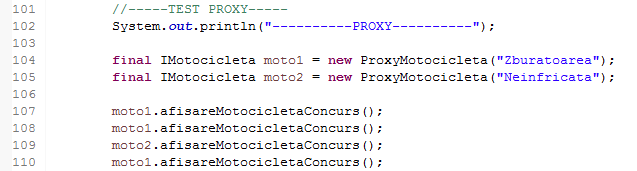


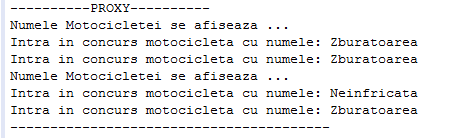
## Proxy

Clase utilizate: Motocicleta, IMotocicleta, ProxyMotocicleta.

Utilizare: Deoarece s-a dorit crearea unei interfețe între diferite framework-uri, s-a recurs la utilizarea caestui Design Pattern.



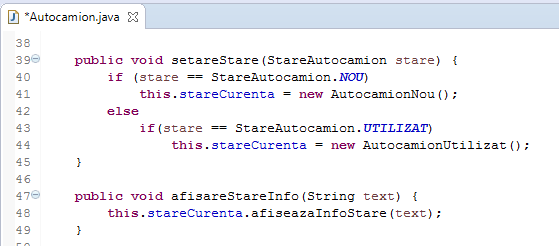


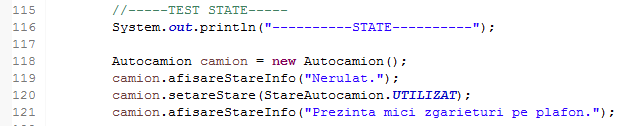


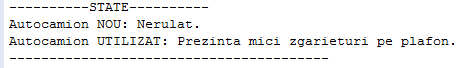
## State

Clase utilizate: Autocamion, IStareAutocamion, StareAutocamion, AutocamionNou, AutocamionUtilizat.

Utilizare: S-a implementat Design Pattern-ul State, deoarece obiectele de tip Autocamion pot avea stările: AutocamionNou, AutocamionUtilizat. Astfel, în funcție de starea unui obiect, aplicația va tratat diferit anumite prelucrări.



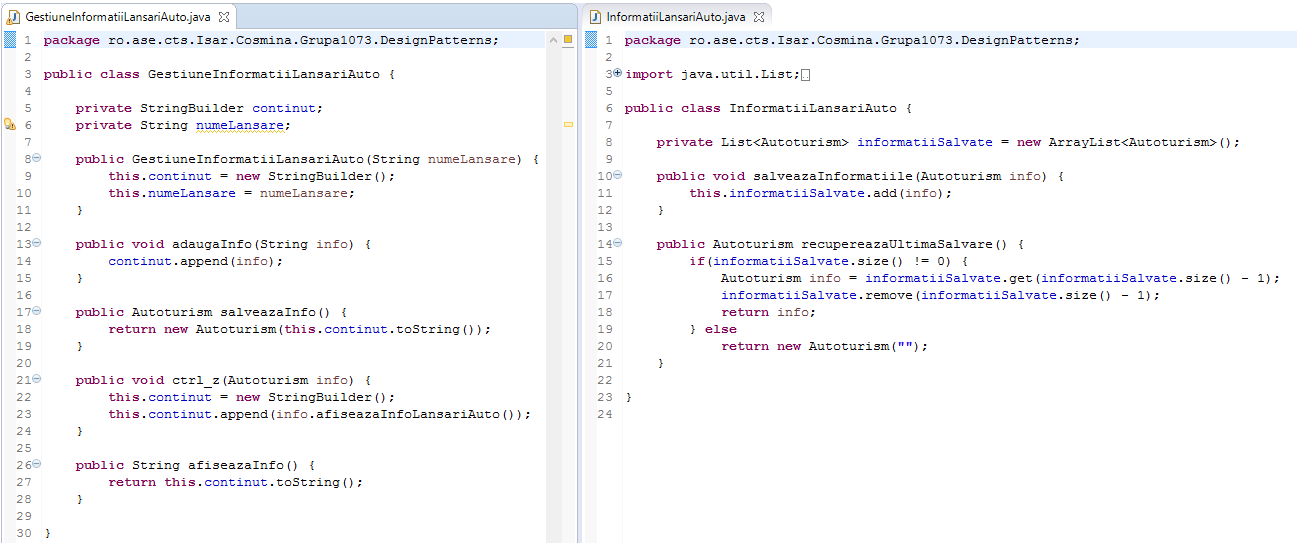


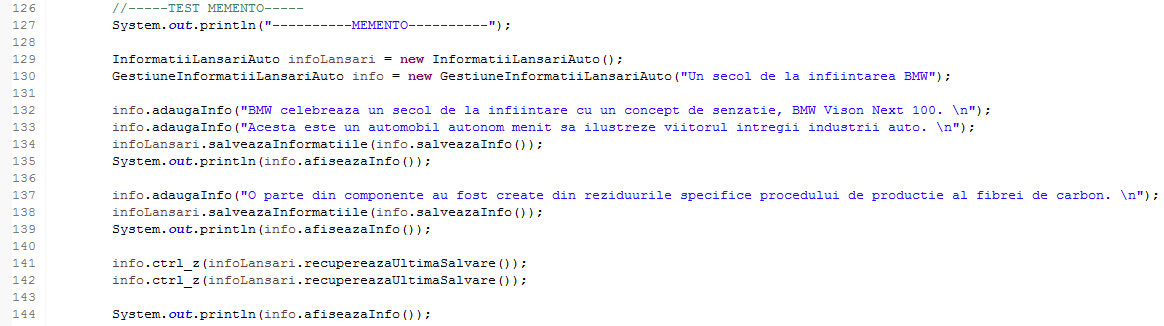


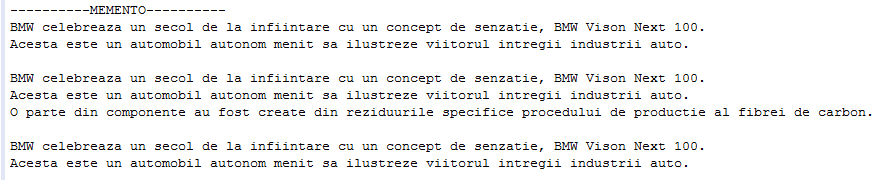
## Memento

Clase utilizate: Autoturism, InformatiiLansariAuto, GestiuneInformatiiLansariAuto.

Utilizare: Prin intermediul aplicării acestui Design Pattern s-a dorit posibilitatea salvării unei stări legate de informațiile aferente unor lansări auto, precum și restaurarea stării pe baza unei imagini anterioare.





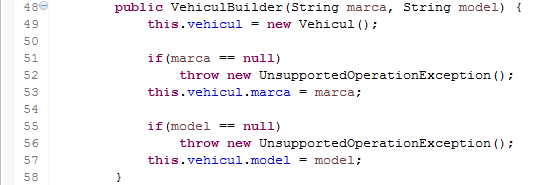


# Definirea și detalierea metodelor testate prin Unit Testing

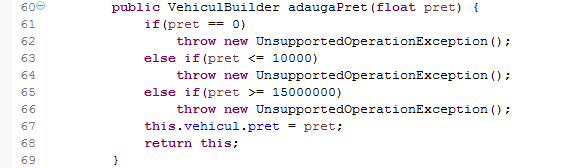
## Clasa Vehicul

Metode testate în cadrul acestei clase:

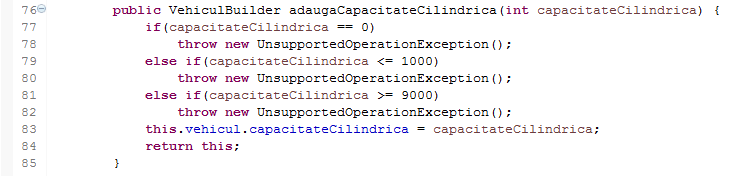
### Interfață comună pentru obiectele de tip Vehicul



### Adăugare componentă preț obiectelor de tip Vehicul



### Adăugare componentă capacitate cilindrică obiectelor de tip Vehicul



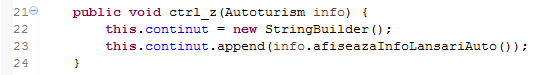
## Clasa GestiuneInformatiiLansariAuto

Metode testate în cadrul acestei clase:

### Adăugare informație privind lansările auto

https://i.gyazo.com/f3158415699ba8b1e8347d00bec680fd.png

### Restaurare informație anterioare

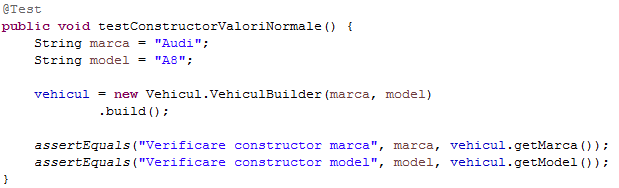


# Definirea și descrierea Test Case-urilor

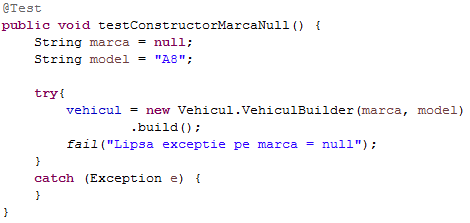
## TestCaseConstructorVehicul

Teste realizate la nivelul acestei metode:

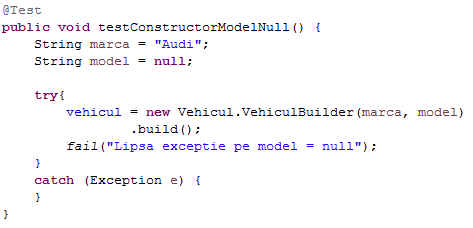
### Verificarea constructorului pentru parametri cu valori normale



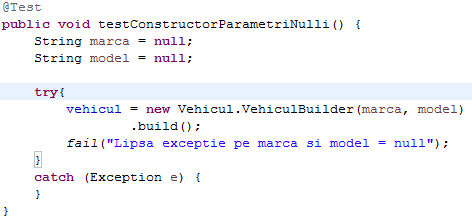
### Verificarea constructorului pentru parametrul marca = null



### Verificarea constructorului pentru parametrul model = null



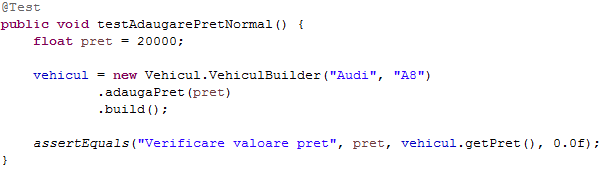
### Verificarea constructorului pentru ambi parametri = null



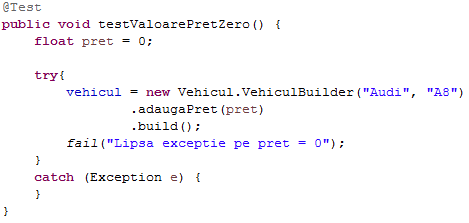
## TestCaseAdaugarePret

Teste realizate la nivelul acestei metode:

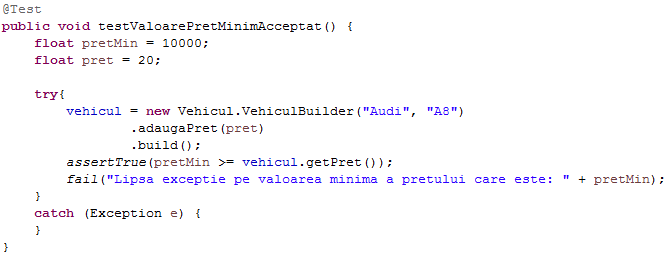
### Verificarea componentei de adaugare pret pentru parametru cu valori normale



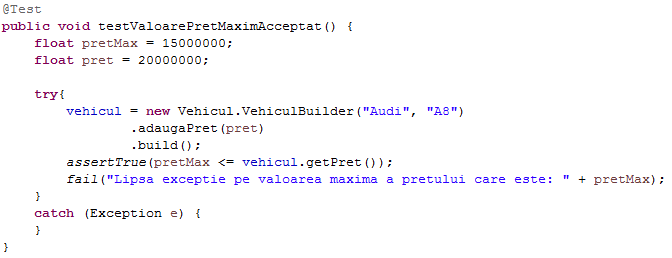
### Verificarea componentei de adaugare pret pentru parametru cu valoare egală cu 0



### Verificarea componentei de adaugare pret pentru parametru cu valoare mai mică decât valoarea minimă acceptată



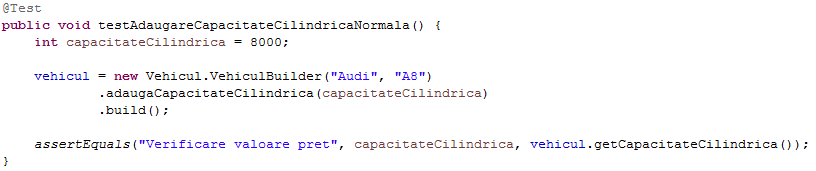
### Verificarea componentei de adaugare pret pentru parametru cu valoare mai mare decât valoarea maximă acceptată



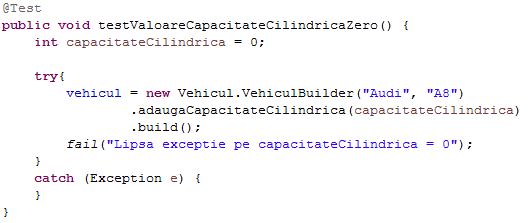
## TestCaseAdaugareCapacitateCilindrica

Teste realizate la nivelul acestei metode:

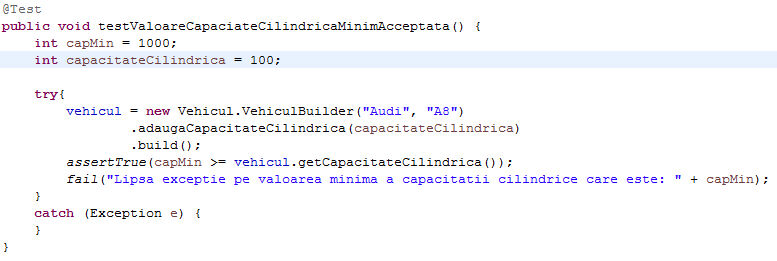
### Verificarea componentei de adaugare capacitate cilindrică pentru parametru cu valori normale



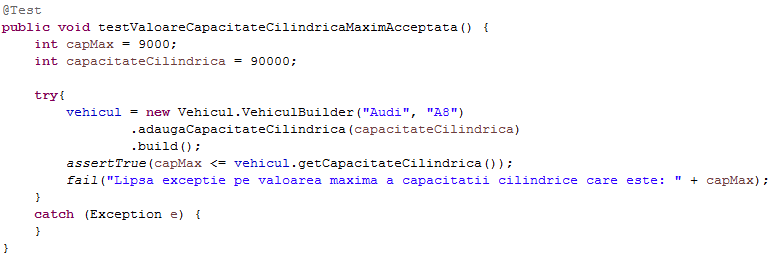
### Verificarea componentei de adaugare capacitate cilindrică pentru parametru cu valoare egală cu 0



### Verificarea componentei de adaugare capacitate cilindrică pentru parametru cu valoare mai mică decât valoarea minimă acceptată



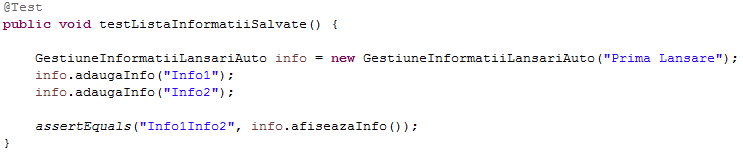
### Verificarea componentei de adaugare capacitate cilindrică pentru parametru cu valoare mai mare decât valoarea maximă acceptată



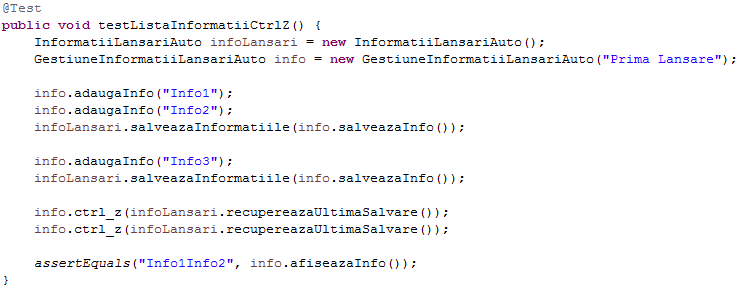
## TestCaseListaInformatiiSalvate

Teste realizate la nivelul acestei metode:

### Verificare metodă de adăugare a unei noi informații privind lista cu informațiile aferente lansărilor auto



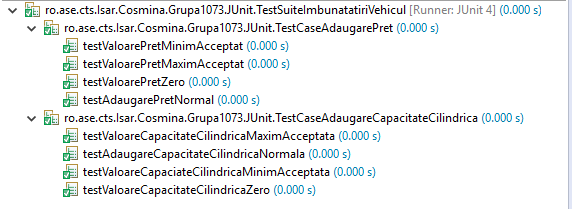
### Verificarea metodei de restaurare a unei informații anterioare din lista cu informațiile aferente lansărilor auto



# Definirea și descrierea Test Suite-urilor

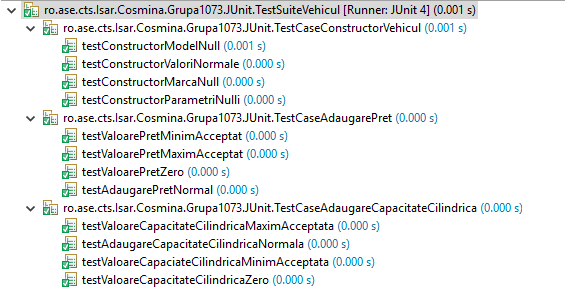
## TestSuiteImbunatatiriVehicul

În cadrul acestui Test Suite au fost adăugate următoarele Test Case-uri:



## TestSuiteVehicul

În cadrul acestui Test Suite au fost adăugate următoarele Test Case-uri:



# Descrierea sumară a funcțiilor aplicației cu referire la pattern-uri și la metodele testate

Această aplicație a fost creată pentru a evidenția implementarea celor opt Design Pattern-uri: Builder, Simple Factory, Adapter, Facade, Decorator, Proxy, State, Memento.

Prin intermediul aplicației pot fi gestionate diverse funcționalități la nivelul celor trei tipuri de Vehicul: Autoturism, Motocicletă, Autocamion. Acestea au o serie de caracteristici comune gestionate prin intermediul clasei Vehicul care sunt obligatorii și care trebuie să respecte un anume intervat, precum și o serie de caracteristici care pot fi adăugate în funcție de necesitate/preferințe. În ceea ce privește obiectele de tip Autoturism pot avea diverse îmbunătățiri, unele oferite prin intermediul unui pachet standard, iar altele prin intermediul unui pachet premium. Obiectele de tip Autocamion au o caracteristică suplimentară, și anume starea. Acestea pot fi ori noi ori utilizate.

Deoarece se dorește o informare cu privire la anumite lansări auto, s-a realizat o gestiune a acestor informații.

# Bibliografie

* Cursuri
* Seminarii